



This application is submitted in the name of the following inventor(s):

<i>Inventor</i>	<i>Citizenship</i>	<i>Residence City and State</i>
Stephane KASRIEL	France	San Francisco, CA

The assignee is *Fireclick, Inc.*, a corporation having an address at 289 South San Antonio Road, Los Altos, CA 94022.

TITLE OF THE INVENTION

Delta Caching Service

BACKGROUND OF THE INVENTION

1. *Field of the Invention*

The invention relates to a delta caching service, and related methods and systems.

2. *Related Art*

In a system that delivers information from a server to clients who request that information (such as a web server delivering information to a set of web clients), it is desirable to minimize the amount of data that is actually sent from the web server to the web client. Delta

1 caching is a technique by which the server and the client differentiate between template informa-
2 tion and delta information for an object to be delivered from the server to the client; the client
3 maintains a copy of the template information and the server is therefore able to deliver the object
4 by only sending the delta information.

5
6 The web server often manages relatively large loads by dividing its tasks, using a
7 load balancer, among a set of server-responders. However, differing server-responders might
8 then associate different template information with objects at the server. One consequence is that
9 when a client communicates with more than one server-responder, the template information
10 known to the client and to the server-responder might differ. This is increasingly more likely as
11 the number of server-responders is increased, and might result in miscommunication between a
12 client and one or more of the server-responders.

13 14 SUMMARY OF THE INVENTION

15
16 The invention provides a method and system capable of ensuring that each client
17 can consistently communicate with one or more servers using delta caching. Building templates
18 for objects at the server(s) is functionally separated from encoding objects for delivery to clients.
19 One or more template-builders are logically separated from one or more delta-encoders. Each
20 operates independently to perform its part of delta caching; template-builders build templates,
21 while delta-encoders use those templates to encode objects for delta caching.

1 In an aspect of the invention, template-builders also operate so that those clients
2 not configured for explicit delta caching can perform implicit ("clientless") delta caching by ref-
3 erence to templates maintained at one or more template-builders. In an aspect of the invention,
4 delta-encoders also operate so that the template information and the delta information for any ob-
5 ject can be separately compressed or sent to clients. In one embodiment, delta-encoders operate
6 with template-builders using a client-server technique.

8 BRIEF DESCRIPTION OF THE DRAWINGS

9
10 Figure 1 shows a block diagram of a system including a delta caching service.

11
12 Figure 2 shows a process flow diagram of a method including a delta caching ser-
13 vice.

14 INCORPORATED DISCLOSURES

15
16
17 Inventions described herein can be used in conjunction with technology described
18 in the following documents:

- 19
- 20 • U.S. Patent Application Serial No. 09/734,910, (Express Mail Mailing No. EL 768 961
21 028US), filed December 11, 2000, in the name of Stephane KASRIEL, attorney docket num-
22 ber 155.1002.01, titled "Predictive Pre-download Using Normalized Network Object Identi-
23 fiers", and applications claiming priority therefrom.

- 1 • US Patent Application Serial No. 60/263,247, (Express Mail Mailing No. EK 913558282US,
2 filed January 22, 2001, in the name of Stephane KASRIEL, attorney docket number
3 155.1003.01, titled "Server Driven Differential Caching", and applications claiming priority
4 therefrom.
5
- 6 • U.S. Patent Application Serial No. 10/058,232, (Express Mail Mailing No. EL 734 815 560
7 US), filed October 19, 2001, in the name of Stephane KASRIEL, attorney docket number
8 155.1008.01, titled "Differential Caching with Many-to-One and One-to-Many Mapping",
9 and applications claiming priority therefrom.

10
11 These documents are hereby incorporated by reference as if fully set forth herein,
12 and are sometimes referred to herein as the "incorporated disclosures".
13

14 DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT
15

16 In the description herein, a preferred embodiment of the invention is described,
17 including preferred process steps and data structures. Those skilled in the art would realize, after
18 perusal of this application, that embodiments of the invention might be implemented using a va-
19 riety of other techniques not specifically described, without undue experimentation or further in-
20 vention, and that such other techniques would be within the scope and spirit of the invention.
21
22
23
24

1 *Lexicography*

2

3 The following terms relate or refer to aspects of the invention or its embodiments.

4 The general meaning of each of these terms is intended to be illustrative and in no way limiting.

5

- 6 • **client, server** — In general, these terms refer to devices or software elements operating
7 in a client/server relationship.

8

9 There is no particular requirement that any particular client or any particular server must
10 be a single hardware device or software module. For example, in some embodiments, the
11 client device or the server device might include multiple devices operating cooperatively
12 (such as when networked) or might include a portion of one or more devices.

13

14

15

16

17

18

19

20

21

22

23

In one embodiment, the client includes a personal computer, such as a workstation, a lap-
top, or a handheld computer, having a web browser disposed for requesting web objects
from the server. In the same embodiment, the server includes one or more web servers,
possibly coupled to a network using an element for distributing requests, each of which is
disposed for parsing requests for web objects, and for providing those web objects in re-
sponse thereto. The web objects might be deemed “static,” in which case they are re-
trieved from a file system, a database, or other storage, or might be deemed “dynamic,”
in which case they are at least partially generated by the web server in response to infor-
mation that is possibly time-varying.

1 There is no particular requirement that any particular client or any particular server must
2 be a single hardware device or software module. For example, in some embodiments, the
3 client device or the server device might include multiple devices operating cooperatively
4 (such as when networked) or might include a portion of one or more devices.

- 5
- 6 • **delta caching** — In general, this refers to a technique in which a client obtains informa-
7 tion regarding a web object in at least two parts: “template information,” which refers to
8 information that might be relatively static and is (in one embodiment) retained at the cli-
9 ent, and “delta information,” which refers to changes from the template information and
10 is (in one embodiment) provided by the server in response to a difference between a pre-
11 sent value of the web object and the template information.

- 12
- 13 • **delta encoder** — In general, this refers to an element that computes the delta informa-
14 tion, such as a difference between a present value of the web object and the template in-
15 formation. In one embodiment, the delta encoder might compress the delta information
16 for delivery to the client.

- 17
- 18 • **delta information** — In general, this refers to information to be added to the template in-
19 formation to compose an entire web object.

- 20
- 21 • **encoding web object for delivery** — In general, this refers to a technique in which tem-
22 plate information is identified for the web object, delta information is computed in re-

1 sponse to the web object and the template information, and the delta information is for-
2 matted (such as in an HTTP response message) for delivery to a requesting client.

- 3
- 4 • **implicit (“clientless”) delta caching** — In general, this refers to a delta caching tech-
5 nique in which delta information is formatted for delivery to a requesting client, and in
6 which the formatted delta information includes program fragments (such as JavaScript)
7 stimulating the client to retrieve the template information if that template information is
8 not already at the client.

- 9
- 10 • **template, template information** — In general, this refers to information which is rela-
11 tively static, or information that has not changed since the last request for the same web
12 object, or information retained by the client.

- 13
- 14 • **template builder** — In general, this refers to an element that computes the template in-
15 formation, such as in response to a change in a web object.

- 16
- 17 • **web page, web object** — In general, this refers to an object available at (or dynamically
18 computed at) the server. A web object might include text, pictures, graphics, animation,
19 video or other motion pictures, sound, program fragments or scripts, or other data. When
20 a web object is associated with a specific URL and is intended for presentation by the cli-
21 ent, it might be referred to as a web page.

1 The scope and spirit of the invention is not limited to any of these definitions, or
2 to specific examples mentioned therein, but is intended to include the most general concepts em-
3 bodied by these and other terms.

4 *System Elements*

5
6 Figure 1 shows a block diagram of a system including a delta caching service.

7
8 A system 100 includes a client 110, a communication network 120, a server 130,
9 a delta encoder 140, and a template builder 150.

10 11 Client

12
13 The client 110 includes a workstation having a processor, program and data
14 memory, and mass storage, and is operated by at least one user 111. In one embodiment, the
15 program includes a web browser, disposed for requesting web objects from the server 130, for
16 receiving web objects from the server 130, and for presenting web objects to the user 111. The
17 program and data memory and mass storage collectively include a client cache 112, in which the
18 client 110 records information from the server 130 for presentation to the user 111.

19
20 The client 110 is coupled to the communication network 120, and is disposed for
21 sending request messages 113 to the server 130. In one embodiment, the request messages 113
22 are formatted in a known protocol, such as HTTP (hypertext transfer protocol), SHTTP (secure

1 HTTP), FTP (file transfer protocol), or a variant thereof. The client 110 is also disposed for re-
2 ceiving messages from other elements in the system 100.

4 Network

6 The network 120 includes a communication link capable of delivering informa-
7 tion between the client 110 and other elements in the system 100. In one embodiment, the com-
8 munication network 120 includes an internet. However, in alternative embodiments, the network
9 120 may include an intranet, a LAN or WAN, a portion of a ATM network or PSTN or other
10 switching network, or in general any elements disposed for delivery of information.

12 Server

14 The server 130 includes at least one web server device 131. Each web server de-
15 vice 131 includes a processor, program and data memory, and mass storage. Each web server
16 device 131 is disposed for receiving request messages 113, for retrieving (or dynamically gener-
17 ating) one or more web objects 132 in response thereto, and for formatting a response message
18 133 including those web objects 132.

20 In one embodiment, the server 130 includes a load balancer 134 and a plurality of
21 the web server devices 131. The load balancer 134 includes a processor, and program and data
22 memory, and is disposed for receiving request messages 113 and for delivering those request
23 messages 113 to individual web server devices 131. This has the effect of distributing the work-

1 load of responding to request messages 113 across more than one web server device 131, so the
2 server 130 can respond with relatively less latency to individual request messages 113.

3
4 The web objects 132 might be retrieved from the mass storage, in which case they
5 are deemed “static,” or might be dynamically generated by the server 130 (specifically, by one of
6 the web server devices 131) in response to information that might possibly be time-varying. For
7 example, if the web object 132 specified by one of the request messages 113 includes stock
8 quote information, the server 130 would dynamically generate that web object 132 in response to
9 actual stock quotes (such as retrieved from a separate stock quote server).

10
11 Delta Encoder

12
13 The delta encoder 140 includes a processor, program and data memory, and mass
14 storage, collectively including a request interceptor 141, a request forwarder 142, an object com-
15 parator 143, and a delta formatter 144.

16
17 The request interceptor 141 is coupled to the network 120, and is disposed for in-
18 tercepting request messages 113 that are destined for the server 130.

19
20 The request forwarder 142 is coupled to the network 120 and to the request inter-
21 ceptor 141, and is disposed for forwarding the intercepted request messages 113 to the server
22 130.

1 In alternative embodiments, the server 130 might be disposed for sending all re-
2 sponse messages 133 to the delta encoder 140, in which case the request interceptor 141 and the
3 request forwarder 142 would not be needed. Accordingly, these elements might be regarded as
4 optional.

5
6 The object comparator 143 is coupled to the network 120, and is disposed for re-
7 ceiving response messages 133 from the server 130. The object comparator 143 compares web
8 objects 132 found in those response messages 133, using a set of template information 157 from
9 the template builder 150, and generates delta information 156.

10
11 In one embodiment, the delta encoder 140, from time to time, sends a request to
12 the template builder 150 for the template information 157, and records that template information
13 157 in its memory or mass storage. The delta encoder 140 would therefore have the template in-
14 formation 157 readily available for computing the delta information 156. In alternative embodi-
15 ments, the delta encoder 140 sends a request to the template builder 150 for the template infor-
16 mation 157 in response to need for that information by the object comparator 143, and uses the
17 template information 157 relatively immediately. The delta encoder 140 would therefore have
18 no special requirement for recording that template information 157 in its memory or mass stor-
19 age.

20
21 The delta formatter 144 is coupled to the network 120 and to the object compara-
22 tor 143, and is disposed for formatting the delta information 156 for delivery to the client 110.

1 The delta formatter 144 generates a response message 133 and sends that response message 133
2 to the network 120 for delivery to the client 110.

3
4 In one embodiment, the delta formatter 144 compresses the delta information 156
5 before packaging that information in the response message 133.

6 7 Template Builder

8
9 The template builder 150 includes a processor, program and data memory, and
10 mass storage, collectively including an object requestor 151, a template identifier 152, and a
11 template server 153.

12
13 The object requestor 151 operates, from time to time, to request web objects 132
14 from the server 130. The template builder 150 receives the web objects 132 and records them in
15 its memory or mass storage.

16
17 The template identifier 152 operates, from time to time, on web objects 132 re-
18 corded in memory or mass storage, and generates template information 157. In one embodiment,
19 the template information 157 can be compressed by the template builder 150 for delivery to the
20 delta encoder 140 or to the client 110.

21
22 The template server 153 operates in like manner as the server 130, in that it re-
23 ceives request messages 113 and sends response messages 133. However, the template server

1 153 provides template information 157 instead of web objects 132 in response to the request
2 messages 113.

3
4 The template server 153 is available for responding to requests by the delta en-
5 coder 140, so the delta encoder 140 can obtain template information with which to compute delta
6 information 156.

7
8 In alternative embodiments, the delta encoder 140 can operate to format the delta
9 information 156 using program fragments such as JavaScript, as described in the incorporated
10 disclosures. In this mode of operation, the client 110 does not need to know that the delta infor-
11 mation 156 does not include the entire web object 132. Rather, when the client 110 attempts to
12 present the web object 132 (it has only the delta information 156), the program fragments direct
13 it to request and receive template information 157 from the template server 153, in like manner
14 as it would request and receive a web object 132 from the server 130. Accordingly, the client
15 110 can obtain both the delta information 156 and the template information 157, and it can pre-
16 sent the entire web object 132.

17
18 *Method of Operation*

19
20 Figure 2 shows a process flow diagram of a method including a delta caching ser-
21 vice.

1 A method 200 is performed by the system 100. Although the method 200 is de-
2 scribed serially, the flow points and steps of the method 200 can be performed by separate ele-
3 ments in conjunction or in parallel, whether asynchronously or synchronously, in a pipelined
4 manner, or otherwise. There is no particular requirement that the method 200 must be performed
5 in the same order in which this description lists flow points or steps, except where explicitly so
6 indicated.

7 8 Client

9
10 At a flow point 210, the client 110 is ready to request a web object 132 from the
11 server 130.

12
13 At a step 211, the client 110 sends a request message 113 to the server 130.

14
15 At a step 212, the client 110 receives a response message 133 from the delta en-
16 coder 140, including delta information 156. In one embodiment, the client 110 recognizes the
17 response message 133 as including only delta information 156, and the client 110 knows it must
18 combine the delta information 156 with template information 157 to present the entire web ob-
19 ject 132. In alternative embodiments such as “clientless” delta caching, the response message
20 133 includes at least one program fragment directing the client 110 to obtain the template infor-
21 mation 157 (either from its client cache 111 or from the template server 153).

1 At a step 213, the client 110 determines if it has the requisite template information
2 157 to present the entire web object 132. If not, the client 110 proceeds with the step 214. If so,
3 the client 110 proceeds with the step 215.

4
5 At a step 214, the client 110 requests the requisite template information 157 from
6 the template server 153 or the delta encoder 140. Template information 157 can be served from
7 both the template server 153 or the delta encoder 140 because they both include the template. A
8 client 110 can be instructed to get the template information 157 from either source. Serving the
9 template information 157 from the template server 153 is particularly advantageous in configura-
10 tions in which the server 130 is not associated with a delta encoder 140.

11
12 In other embodiments, a CDN (content delivery network) can be used to serve
13 templates information 157. Thus, there are four possibilities: 1) the template builder 150 serves
14 template information 157 directly to the client 110, (2) the delta encoder 140 serves template in-
15 formation 157 directly to the client 110, (3) the template builder 150 serves template information
16 157 to the client through a CDN and (4) the template builder 150 serves template information
17 157 to the client 110 through a CDN and the delta encoder 140 serves content to a CDN, which
18 serves it to the client 110

19
20 At a step 215, the client 110 combines the delta information 156 with template in-
21 formation 157 to present the entire web object 132.

22
23 The client 110 returns to the flow point 210.

Server

The server 130 operates in like manner as an ordinary web server.

Delta Encoder

At a flow point 220, the delta encoder 140 is ready to encode a web object 132 for delivery to the client 110.

At a step 221, the object comparator 143 receives the web object 132 from the server 130. As noted above, the delta encoder 140, from time to time, requests template information 157 from the template server 153, and records that template information 157 in its memory or mass storage.

At a step 222, the object comparator 143 compares the web object 132 with the template information 157, and generates delta information 156.

At a step 223, the delta formatter 144 formats the delta information 156 for delivery to the client 110, generates a response message 133 and sends that response message 133 to the network 120 for delivery to the client 110. In one embodiment, the delta formatter 144 compresses the delta information 156 before packaging that information in the response message 133.

1 The delta encoder 140 returns to the flow point 220.

2
3 Template Builder

4
5 At a flow point 230, the template builder 150 is ready to build template informa-
6 tion 157. This is done the first time a web object 132 is requested and whenever the delta infor-
7 mation 156 associated with that web object 132 grows “too large”.

8
9 At a step 231, the object requestor 151 requests one of the web objects 132 from
10 the server 130.

11
12 At a step 232, the template builder 150 receives the one web object 132 and re-
13 cords it in its memory or mass storage.

14
15 At a step 233, the template identifier 152 compares the one web object 132 with
16 an earlier version of that web object 132, and generates template information 157 in response to
17 the comparison.

18
19 At an optional step 234, the template identifier 152 compresses the template in-
20 formation 157.

21
22 The template builder 150 returns to the flow point 230.

1 The template server 153 operates in like manner as an ordinary web server, except
2 that it provides template information 157 instead of web objects 132.

3
4 *Alternative Embodiments*

5
6 Although preferred embodiments are disclosed herein, many variations are possi-
7 ble which remain within the concept, scope, and spirit of the invention. These variations would
8 become clear to those skilled in the art after perusal of this application.

- 9
10 • The invention is generally applicable to client-server processes, and to systems in which
11 clients request information and servers provide information in request to responses, such
12 as for example client-server database systems and client-server file systems.

13
14 Those skilled in the art will recognize, after perusal of this application, that these
15 alternative embodiments are illustrative and in no way limiting.